Huber Optimization of Circuits: A Robust Approach

John W. Bandler, *Fellow, IEEE,* Shao Hua Chen, *Member, IEEE,* Radoslaw M. Biernacki, *Senior Member, IEEE,* Li Gao, Kaj Madsen, and Huanyu Yu

***Abstract*—We introduce a novel approach to "robustizing" circuit optimization using Huber functions: both two-sided and one-sided. Advantages of the Huber functions for optimization in the presence of faults, large and small measurement errors, bad starting points, and statistical uncertainties are described. In this context, comparisons are made with optimization using ℓ_1, ℓ_2 and minimax objective functions. The gradients and Hessians of the Huber objective functions are formulated. We contribute a dedicated, efficient algorithm for Huber optimization and show, by comparison, that generic optimization methods are not adequate for Huber optimization. A wide range of significant applications is illustrated, including FET statistical modeling, multiplexer optimization, analog fault location, and data fitting. The Huber concept, with its simplicity and far-reaching applicability, will have a profound impact on analog circuit CAD.**

I. Introduction

Engineering designers are often concerned with the robustness of numerical optimization techniques, and rightly so, knowing that engineering data is, with few exceptions, contaminated by model/measurement/statistical errors.

The classical least-squares (ℓ_2) method is well known for its vulnerability to gross errors: a few wild data points can alter the least-squares solution significantly. The ℓ_1 method is robust against gross errors [1], [2]. We will show, however, that when the data contains many small errors (such as statistical variations), the ℓ_1 solution can be undesirably biased toward a subset of the data points. This indicates that ℓ_1 is not suitable, in general, as a statistical estimator.

Neither the ℓ_2 nor the ℓ_1 method has flexible discriminatory power to recognize and treat differently large (catastrophic) errors and small (soft) errors. We introduce the Huber function [3]-[5], which appears to be a hybrid of the ℓ_1 and ℓ_2 measures. Compared with ℓ_2, the Huber solution is more robust w.r.t. large errors. Compared with ℓ_1, the Huber solution can provide a smoother, less biased estimate from data that contains many small deterministic or statistical variations. We demonstrate the benefits of this novel approach in FET statistical modeling, analog fault location, and data fitting.

We extend the Huber concept by introducing a "one-sided" Huber function for large-scale optimization. For large-scale problems, systematic decomposition techniques have been proposed (e.g., [6], [7]) to reduce computational time and prevent potential convergence problems. In practice, the designer often attempts, by intuition, a "preliminary" optimization with a small number of dominant variables. The full-scale optimization is performed if and when a reasonably good point is obtained.

With a reduced number of variables, the optimizer may not be able to reduce all the error functions at the same time. For instance, the specification may be violated more severely at some sample points (such as frequencies) than at the others. In such situations, the minimax method is preoccupied with the worst-case errors and therefore becomes ineffective or inefficient. We demonstrate, through microwave multiplexer optimization, that the one-sided Huber function can be more effective and efficient than minimax in overcoming a bad starting point.

We present a dedicated, efficient, gradient-based algorithm for Huber optimization and show, by comparison, that generic optimization methods, such as quasi-Newton, conjugate gradient, and simplex algorithms, are not adequate when directly applied to minimizing the Huber objective functions. The gradients and Hessians of the Huber objective functions are derived and their significance is discussed.

II. Theoretical Formulation of Huber Functions

The Huber optimization problem is defined as [3], [4]

$$\underset{x}{\text{minimize}}\ F(\boldsymbol{x}) \triangleq \sum_{j=1}^{m} \rho_k(f_j(\boldsymbol{x})) \tag{1}$$

where $\boldsymbol{x} = [x_1 \ x_2 \ \cdots \ x_n]^T$ is the set of variables and ρ_k is the Huber function defined as

$$\rho_k(f) = \begin{cases} f^2/2 & \text{if } |f| \le k \\ k|f| - k^2/2 & \text{if } |f| > k \end{cases} \tag{2}$$

where k is a positive constant and f_j, $j = 1, 2, \cdots, m$, are error functions.

The Huber function ρ_k is a hybrid of the least-squares

Manuscript received March 29, 1993; revised June 16, 1993. This work was supported in part by Optimization Systems Associates Inc., and in part by the Natural Sciences and Engineering Research Council of Canada under Grants OGP0007239, OGP0042444, and STR0117819.

J. W. Bandler, S. H. Chen, and R. M. Biernacki are with Optimization Systems Associates Inc., P.O. Box 8083, Dundas, Ont., Canada L9H 5E7, and the Simulation Optimization Systems Research Laboratory, Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada L8S 4L7.

L. Gao is with the Department of Mathematics and Institute of Mathematics, Peking University, Beijing 100871, China.

K. Madsen is with the Institute for Numerical Analysis, The Technical University of Denmark, DK-2800 Lyngby, Denmark.

H. Yu is with the Simulation Optimization Systems Research Laboratory, Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada L8S 4L7.

IEEE Log Number 9213017.

0018-9480/93$03.00 © 1993 IEEE

(ℓ_2) (when $|f| \leq k$) and the ℓ_1 (when $|f| > k$) functions. As illustrated in Figs. 1 and 2, the definition of ρ_k ensures a smooth transition between ℓ_2 and ℓ_1 at $|f| = k$. This means that the first derivative of ρ_k w.r.t. f is continuous.

The ℓ_1 is robust against gross errors in the data [1], [2]. Since the Huber function treats errors above the threshold (i.e., $|f| > k$) in the ℓ_1 sense, it is robust against those errors, i.e., the solution is not sensitive to those errors. The choice of k defines the threshold between "large" and "small" errors. By varying k, we can alter the proportion of error functions to be treated in the ℓ_1 or ℓ_2 sense. Huber gave a look-up table [3] from which k can be determined according to the percentage of gross errors in the data. If k is set to a sufficiently large value, the optimization problem (1) becomes least squares. On the other hand, as k approaches zero, ρ_k/k will approach the ℓ_1 function.

A. Gradient and Hessian

To further our insight into the properties of the Huber formulation, we derive the gradient and Hessian of the Huber objective function as follows.

The gradient vector of the Huber objective function F w.r.t. $\boldsymbol{x}$ is given by

$$\nabla F = \sum_{j=1}^{m} v_j f_j' \tag{3}$$

where

$$v_j \triangleq \frac{\partial \rho_k(f_j(\boldsymbol{x}))}{\partial f_j(\boldsymbol{x})} = \begin{cases} f_j(\boldsymbol{x}) & \text{if } |f_j(\boldsymbol{x})| \leq k \\ \pm k & \text{if } |f_j(\boldsymbol{x})| > k \end{cases} \tag{4}$$

$$f_j' \triangleq \left[\frac{\partial f_j(\boldsymbol{x})}{\partial x_1} \; \frac{\partial f_j(\boldsymbol{x})}{\partial x_2} \cdots \frac{\partial f_j(\boldsymbol{x})}{\partial x_n} \right]^T. \tag{5}$$

The structure of (3) is very similar to the gradient of ℓ_2 (least squares), which is

$$\nabla F_{\ell_2} = \sum_{j=1}^{m} f_j f_j'. \tag{6}$$

By comparing (3) with (6), we can see that v_j, namely the first derivative of ρ_k w.r.t. f_j, serves as a weighting factor in the Huber gradient. For $|f_j| \leq k$, v_j is defined in (4) as f_j, which is the same as in the ℓ_2 gradient given by (6). For $|f_j| > k$, v_j is held constant at the value of f_j at the threshold. In other words, the Huber gradient can be thought of as a modified ℓ_2 gradient, where the gross errors are reduced to the threshold value.

The Hessian matrix of the Huber objective function F w.r.t. $\boldsymbol{x}$ can be expressed as

$$\boldsymbol{H} = \sum_{j=1}^{m} (d_j f_j' f_j'^T + v_j f_j'') \tag{7}$$

where

$$d_j \triangleq \frac{\partial^2 \rho_k(f_j(\boldsymbol{x}))}{\partial f_j^2(\boldsymbol{x})} = \begin{cases} 1 & \text{if } |f_j(\boldsymbol{x})| \leq k \\ 0 & \text{if } |f_j(\boldsymbol{x})| > k \end{cases} \tag{8}$$

$$f_j'' \triangleq \frac{\partial f_j'^T}{\partial \boldsymbol{x}}. \tag{9}$$

Comparing (7) with the ℓ_2 Hessian matrix given by

$$\boldsymbol{H}_{\ell_2} = \sum_{j=1}^{m} (f_j' f_j'^T + f_j f_j'') \tag{10}$$

we can see that v_j serves as a weighting factor to reduce the contribution of gross errors in the data to the Hessian matrix.

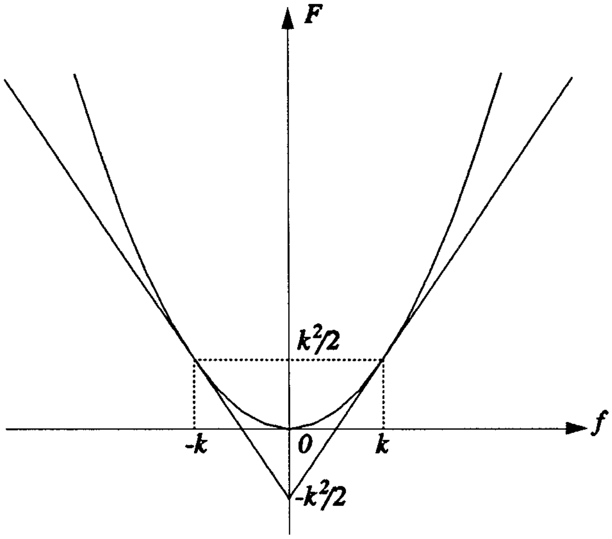


Fig. 1. The ℓ_1 and ℓ_2 objective functions in the one-dimensional case. The ℓ_1 function is rescaled and shifted in accordance with the corresponding part in the Huber function. It has the form $F = k|f| - k^2/2$. The ℓ_2 function has the form $F = f^2/2$.

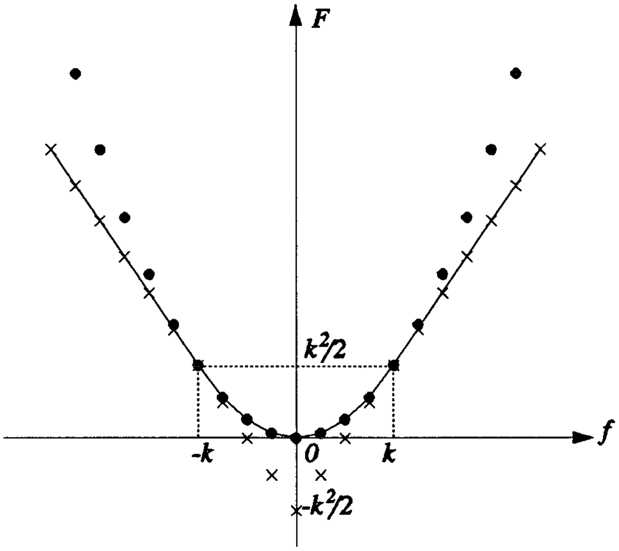


Fig. 2. The Huber, ℓ_1 and ℓ_2 objective functions in the one-dimensional case. The strikes and dots represent the discrete points on the ℓ_1 and ℓ_2 curves, respectively, in Fig. 1. The continuous curve indicates the Huber objective function.

B. One-Sided Huber Function

We present an extension of the Huber concept by introducing the "one-sided" Huber optimization defined as

$$\underset{x}{\text{minimize}}\ F(\boldsymbol{x}) \triangleq \sum_{j=1}^{m} \rho_k^+(f_j(\boldsymbol{x})) \tag{11}$$

where

$$\rho_k^+(f) = \begin{cases} 0 & \text{if } f \le 0 \\ f^2/2 & \text{if } 0 < f \le k \\ kf - k^2/2 & \text{if } f > k. \end{cases} \tag{12}$$

This one-sided Huber function is tailored for design optimization with upper and/or lower specifications. f is truncated when negative because the corresponding design specification is satisfied.

The gradient vector of the one-sided Huber objective function F w.r.t. $\boldsymbol{x}$ is given by

$$\nabla F = \sum_{j=1}^{m} v_j^+ \boldsymbol{f}_j' \tag{13}$$

where

$$v_j^+ \triangleq \frac{\partial \rho_k^+}{\partial f_j} = \begin{cases} 0 & \text{if } f_j \le 0 \\ f_j & \text{if } 0 < f_j \le k \\ k & \text{if } f_j > k. \end{cases} \tag{14}$$

The Hessian matrix of the one-sided Huber objective function is given by

$$\boldsymbol{H} = \sum_{j=1}^{m} (d_j^+ \boldsymbol{f}_j' \boldsymbol{f}_j'^T + v_j^+ \boldsymbol{f}_j'') \tag{15}$$

where

$$d_j^+ = \frac{\partial^2 \rho_k^+}{\partial f_j^2} = \begin{cases} 0 & \text{if } f_j \le 0 \\ 1 & \text{if } 0 < f_j \le k \\ 0 & \text{if } f_j > k. \end{cases} \tag{16}$$

III. A Dedicated Algorithm for Huber Optimization

We present a dedicated, efficient algorithm for minimizing the Huber objective functions, both one- and two-sided. We have implemented this algorithm in the CAD system OSA90/hope$^{\text{TM}}$ [8] as a new standard feature and used it to generate the numerical results presented in this paper.

The numerical algorithms proposed for solving (1) are of the trust region type. We calculate a sequence of points $\{\boldsymbol{x}_p\}$ intended to converge to a local minimum of F. At each iterate $\boldsymbol{x}_p$, a linear function l_j is used to approximate the nonlinear function f_j, $j = 1, 2, \cdots, m$, and thus a linearized model L_p of F is constructed. This model is a good approximation to F within a specified neighborhood N_p of the pth iterate $\boldsymbol{x}_p$. This neighborhood N_p is intended to reflect the domain in which the l_j approximations of the f_j are valid.

Assume a tentative step $\boldsymbol{h}$ is being searched at the pth iterate $\boldsymbol{x}_p$. If the search is successful, we go on to the next iteration, i.e., $\boldsymbol{x}_{p+1} = \boldsymbol{x}_p + \boldsymbol{h}$. The problem is formulated as

$$\underset{h}{\text{minimize}}\ L_p(\boldsymbol{h}) \triangleq L(\boldsymbol{h}, \boldsymbol{x}_p) = \sum_{j=1}^{m} \rho_k(l_j(\boldsymbol{h}, \boldsymbol{x}_p)) \tag{17}$$

where

$$l_j(\boldsymbol{h}, \boldsymbol{x}_p) \triangleq f_j(\boldsymbol{x}_p) + [\boldsymbol{f}_j'(\boldsymbol{x}_p)]^T \boldsymbol{h} \tag{18}$$

subject to the constraint $\boldsymbol{h} \in N_p$, where

$$N_p \triangleq \{\boldsymbol{x} |\ \|\boldsymbol{x} - \boldsymbol{x}_p\| \le \delta_p\} \tag{19}$$

and where $\|\cdot\|$ denotes the Euclidean norm.

The difference between the Hessians of the true Huber objective function (7) and this linearized model is the term

$$\sum_{j=1}^{m} v_j \boldsymbol{f}_j''.$$

This error in approximating the true Hessian (7) is smaller than in the ℓ_2 case, namely,

$$\sum_{j=1}^{m} f_j \boldsymbol{f}_j''.$$

We solve the foregoing problem (17) using an algorithm similar to that of Madsen and Nielsen for the linear Huber problem [9]. This method is based on the fact that L_p is a combination of quadratic functions which are linked together in a smooth manner. Therefore, a Newton iteration is very efficient, and can be proved to find the solution after a finite number of steps. The solution to this linear problem is denoted by $\boldsymbol{h}_p$.

The trust region radius δ_p is updated in each iteration. We propose the usual updating scheme for trust region methods (e.g., see Moré [10]). This is based on the ratio

$$r_p = \frac{F(\boldsymbol{x}_p) - F(\boldsymbol{x}_p + \boldsymbol{h}_p)}{L_p(\boldsymbol{0}) - L_p(\boldsymbol{h}_p)} \tag{20}$$

i.e., the ratio between the decrease in the nonlinear function and the decrease in the local approximation. If r_p is close to 1 then we can afford a larger trust region in the next iteration. On the other hand, if r_p is too small, the the trust region must be decreased.

The new point $\boldsymbol{x}_p + \boldsymbol{h}_p$ is only accepted if the objective function F decreases by a factor no less than s_1. Otherwise, another tentative step is calculated from $\boldsymbol{x}_p$ using a decreased trust region. A more precise step-by-step description of the algorithm follows.

Step 1: Given $\boldsymbol{x}_0$ and $\delta_0 > 0$. Let $0 < s_2 < 1 < s_3$. (These constants are chosen according to our experience. The algorithm is not sensitive to small changes in these constants.) Set the iteration count $p = 1$.

Step 2: Solve the trust region linearized subproblem to find the minimizer $\boldsymbol{h}_p$ of (17) subject to (19).

Step 3: If $F(\boldsymbol{x}_p + \boldsymbol{h}_p) < (1 - s_1)F(\boldsymbol{x}_p)$, let $\boldsymbol{x}_{p+1} = \boldsymbol{x}_p + \boldsymbol{h}_p$; otherwise let $\boldsymbol{x}_{p+1} = \boldsymbol{x}_p$.

Step 4: If $r_p \le 0.25$, reduce the size of the trust region

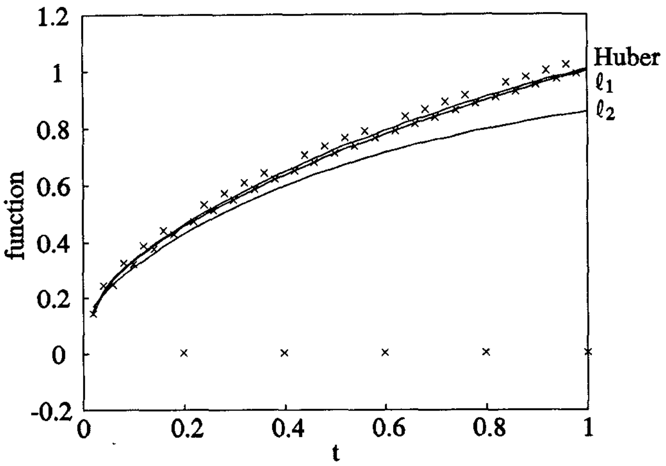


Fig. 3. ℓ_1, ℓ_2 and Huber solutions for data fitting in the presence of errors.

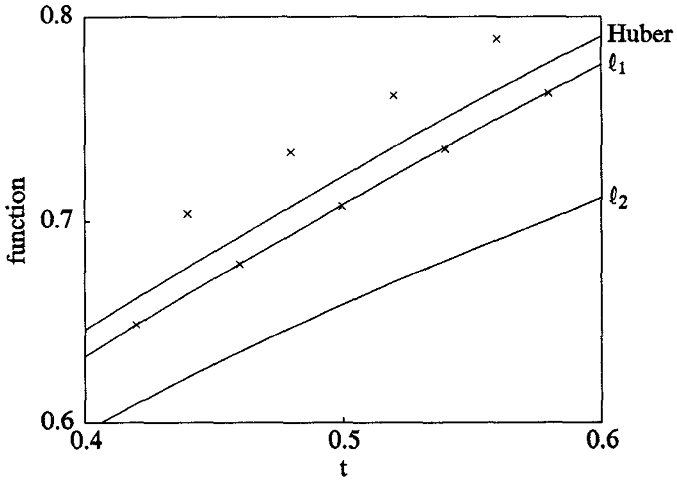


Fig. 4. An enlarged portion of Fig. 3.

by letting $\delta_{p+1} = \delta_p s_2$; or if $r_p \geq 0.75$, increase the size of the trust region by letting $\delta_{p+1} = \delta_p s_3$; otherwise keep the trust region size unchanged by letting $\delta_{p+1} = \delta_p$.

Step 5: If the convergence criteria are satisfied, stop; otherwise update the iteration count by letting $p = p + 1$ and repeat from Step 2.

It has been proved in [4] that this algorithm obeys the usual convergence theory for trust region methods.

IV. Comparison of ℓ_1, ℓ_2 and Huber Methods in Data Fitting

To illustrate the characteristics of the ℓ_1, ℓ_2 and Huber solutions for data fitting problems in the presence of large and small errors, we consider the approximation of $\sqrt{t}$ by the rational function

$$F(x, t) = \frac{x_1 t + x_2 t^2}{1 + x_3 t + x_4 t^2} \qquad (21)$$

for $0 \leq t \leq 1$ [2]. $\sqrt{t}$ is uniformly sampled at 0.02, 0.04, $\cdots$, 1. We deliberately introduced large errors at 5 of the sample points and small variations to the remaining data. The ℓ_1, ℓ_2 and Huber solutions are obtained by optimizing the coefficients x_1, x_2, x_3, and x_4 in (21) to match the sampled data using the respective objective functions. The results are shown in Fig. 3. A portion of Fig. 3 is enlarged in Fig. 4 for a clearer view of the details.

As expected, the least-squares solution suffers significantly from the presence of the five erroneous points. On the other hand, the ℓ_1 solution, according to the optimality condition, is dictated by a subset of residual functions which have zero values at the solution. In a sense, all the nonzero residuals are viewed as large errors. This tendency towards a biased ℓ_1 solution, as dramatized in our example, is undesirable if we wish to model the small variations in the data.

The Huber solution features a flexible combination of the robustness of the ℓ_1 and the unbiasedness of the ℓ_2. In fact, the Huber solution is equivalent to an ℓ_2 solution with the gross errors reduced to the threshold value k. In our example, k is chosen as 0.04 according to the magnitude of the small variations in the data.

V. Huber Estimator for Statistical Modeling of Devices

One approach to statistical modeling of devices [11]–[13] is to extract the model parameters from a sample of device measurements and then postprocessing the sample of model parameters to estimate their statistics (means, statistical deviations, and correlations).

To estimate the mean of a parameter by optimization, we define the error functions as

$$f_j(\overline{\phi}) = \overline{\phi} - \phi^j, \qquad j = 1, 2, \cdots, N \qquad (22)$$

where ϕ^j is the extracted parameter value for the jth device and N is the total number of devices. Similarly, to estimate the variances, we define

$$f_j(V_\phi) = V_\phi - (\phi^j - \overline{\phi})^2, \qquad j = 1, 2, \cdots, N \qquad (23)$$

where V_ϕ denotes the estimated variance from which we can calculate the standard deviation σ_ϕ. The model parameters we use are extracted from the measurements of 80 FETs [14].

When the postprocessing is done using a least-squares estimator, problems will arise if the measurements contain gross measurement errors and/or involve faulty devices. For example, consider the run chart shown in Fig. 5 of an extracted model parameter, namely the FET time-delay τ.

Most of the extracted values of τ are between 2 and 2.5 ps, but there are a few abnormal values due to faulty devices and/or gross measurement errors. These wild points will severely affect the ℓ_2 estimator. In fact, the other model parameters extracted from those faulty devices also have abnormal values. In our earlier work [11], [12] using the ℓ_2 estimator, the abnormal data sets were manually excluded from the statistical modeling process.

The Huber function can be used as an automatic robust statistical estimator. The threshold value k is chosen to reflect the normal spread of the parameter values (e.g., we chose $k = 0.25$ ps for τ).

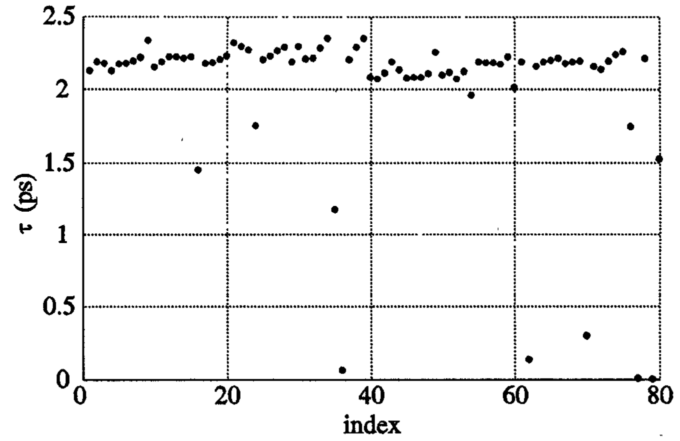


Fig. 5. Run chart of the extracted FET time-delay τ.

TABLE I
ESTIMATED STATISTICS OF SELECTED FET PARAMETERS

Parameter[a]	$\bar{\phi}$ (ℓ_2)	$\bar{\phi}$ (Huber)	$\bar{\phi}$ (ℓ_2^*)[b]	σ_ϕ (ℓ_2)	σ_ϕ (Huber)	σ_ϕ (ℓ_2^*)[b]
L_G (nH)	0.04387	0.03464	0.03429	94.6 percent	21.8 percent	17.4 percent
G_{DS} (1/KΩ)	1.840	1.820	1.839	28.6 percent	6.3 percent	4.9 percent
I_{DSS} (mA)	47.36	47.53	47.85	14.0 percent	12.7 percent	11.3 percent
τ (ps)	2.018	2.154	2.187	26.3 percent	5.8 percent	3.4 percent
C_{10} (pF)	0.3618	0.3658	0.3696	8.2 percent	4.6 percent	3.5 percent
K_1	1.2328	1.231	1.233	15.5 percent	10.8 percent	8.7 percent

[a] L_G represents the FET gate lead inductance, G_{DS} the drain-source conductance, I_{DSS} the drain saturation current, τ the time-delay, C_{10} and K_1 are parameters in the definition of the gate nonlinear capacitor.

[b] ℓ_2^* denotes ℓ_2 estimates after 11 abnormal sets are manually excluded [11].

Table I lists the means and standard deviations of a selected number of model parameters we have obtained using the ℓ_2 and the Huber estimators (the Materka and Kacprzak FET model [15] is used). For comparison, we also list the results obtained using the ℓ_2 estimator *after the abnormal data sets are manually excluded.*

The impact of the abnormal data points on the ℓ_2 estimates of the standard deviations is especially severe. Compared with ℓ_2^*, the Huber estimator does not require manual manipulation of the data and is more appropriate when there are data points that cannot be clearly classified as normal or abnormal.

It should also be noted that although ℓ_1 is effective for individual device parameter extraction, it is not, in general, suitable for statistical postprocessing. The ℓ_1 estimate (median) depends on the order rather than the actual values of the sample.

To illustrate the dependence of the Huber estimates on the threshold k, we list in Table II the estimated statistics of the parameter τ for different values of k. We can also define N_s as the number of "small errors," i.e., the cardinality of the set $\{f_j \mid |f_j| \leq k\}$, at the solution of Huber optimization for each value of k. Fig. 6 depicts N_s versus k, where N_s is expressed as a percentage of the total number of devices N. The "knee" on the curve corresponds to a solution that includes a majority of functions as "small errors." The value of k at the "knee" is consistent with our choice. Figs. 7 and 8 depict N_s for two other parameters, namely L_G and C_{10}, respectively.

TABLE II
ESTIMATED STATISTICS FOR DIFFERENT VALUES OF k

k	$\bar{\tau}$	σ_τ
0.15	2.168	4.4 percent
0.2	2.161	5.1 percent
0.225	2.157	5.4 percent
0.25	2.154	5.8 percent
0.275	2.150	6.2 percent
0.3	2.147	6.6 percent
0.5	2.122	9.6 percent
1	2.079	15.7 percent
∞	2.018	26.3 percent

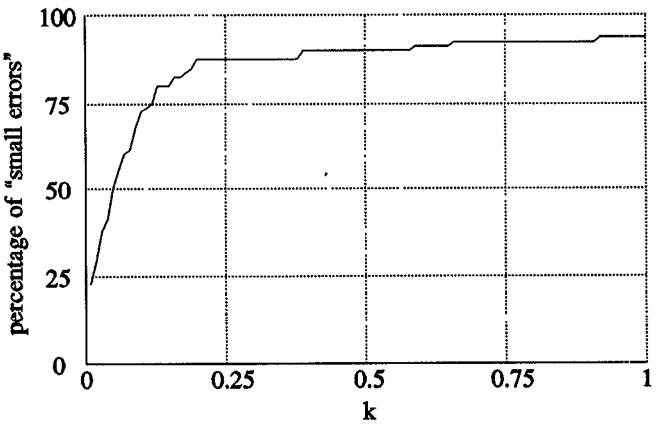


Fig. 6. Percentage of "small errors" for the FET time-delay τ versus the threshold k.

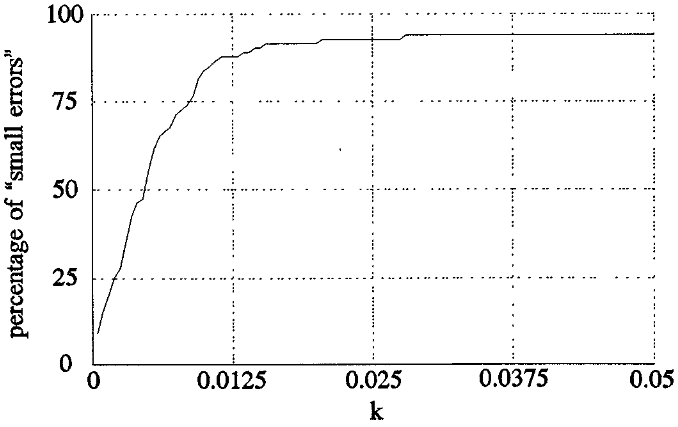


Fig. 7. Percentage of "small errors" for the FET gate lead inductance L_G versus the threshold k.

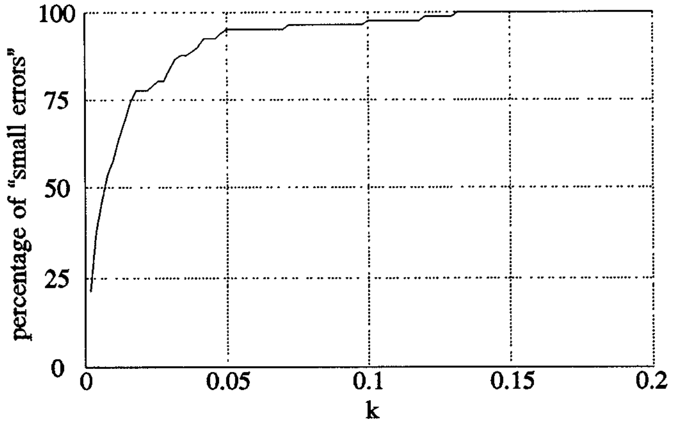


Fig. 8. Percentage of "small errors" for the FET model parameter C_{10} versus the threshold k.

VI. Application to Analog Fault Location

The ℓ_1 method has been applied successfully to the problem of fault location in analog circuits [1], [16], [17]. Typically, a faulty circuit contains only a few faults and possibly many small tolerances for the other elements. Also, the measurements taken on the faulty circuit are usually insufficient for complete parameter identification and, therefore, a robust optimization procedure is needed.

The fault location problem can be formulated as the ℓ_1 optimization [1]

$$\underset{x}{\text{minimize}} \sum_{i=1}^{n} |\Delta x_i / x_i^0| \tag{24}$$

subject to

$$V_1^c - V_1^m = 0$$

$$\vdots$$

$$V_K^c - V_K^m = 0$$

where $x = [x_1 \;\; x_2 \;\; \cdots \;\; x_n]^T$ is a vector of circuit parameters and x^0 represents the nominal parameter values. $\Delta x_i = x_i - x_i^0$ represents the deviation of the ith parameter from its nominal value. $V_1^m, \cdots, V_K^m$ are K measurements on the circuit under test (e.g., voltages measured at accessible nodes under one or more excitations). $V_1^c, \cdots, V_K^c$ are the calculated circuit responses.

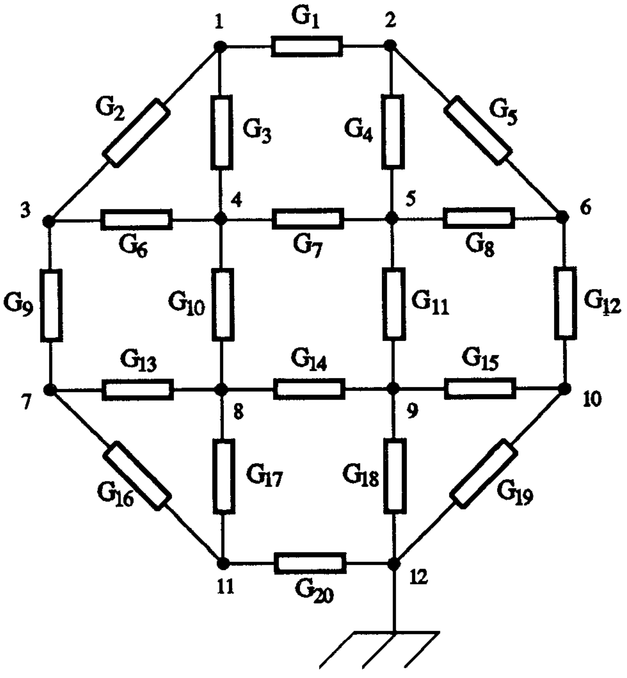


Fig. 9. The resistive mesh circuit.

Instead of the constrained optimization problem (24) we use the Huber method to minimize the following penalty function

$$\underset{x}{\text{minimize}} \sum_{j=1}^{n+K} \rho_k(f_j(x)) \tag{25}$$

where

$$f_i(x) = \Delta x_i / x_i^0, \qquad i = 1, 2, \cdots, n$$

$$f_{n+i}(x) = \beta_i (V_i^c - V_i^m), \qquad i = 1, 2, \cdots, K \tag{26}$$

and β_j, $i = 1, 2, \cdots, K$, are appropriate multipliers for the penalty terms.

Consider the resistive mesh network shown in Fig. 9 [1], [16]. The nominal element values are $G_i = 1.0$ with tolerances $\epsilon_i = \pm 0.05$, $i = 1, 2, \cdots, 20$. Node 12 is taken as the reference node, and nodes 4, 5, 8, and 9 are assumed to be internal and inaccessible for measurement. The voltage measurements at the other nodes are used for fault location.

The actual parameter values of a faulty network are listed in Table III. Two faults are assumed in the circuit, namely G_2 and G_{18}. A single excitation (a dc current source) is applied to node 1. Simulated voltage measurement data is obtained by circuit simulation using the actual parameter values. The nominal parameter values are used as the starting point for optimization. The results from the ℓ_1 optimization and Huber optimization are compared in Table III. The threshold k for the Huber function is chosen as 0.05, commensurate with the tolerances of

TABLE III
FAULT LOCATION OF THE RESISTIVE MESH CIRCUIT

Element	Nominal Value	Actual Value	Percentage Deviation		
			Actual	ℓ_1	Huber
G_1	1.0	0.98	−2.0	0.00	−0.11
G_2	1.0	0.50	−50.0[a]	−48.89	−47.28
G_3	1.0	1.04	4.0	0.00	−2.46
G_4	1.0	0.97	−3.0	0.00	−1.18
G_5	1.0	0.95	−5.0	−2.70	−3.16
G_6	1.0	0.99	−1.0	0.00	−0.06
G_7	1.0	1.02	2.0	0.00	−0.19
G_8	1.0	1.05	5.0	0.00	−0.41
G_9	1.0	1.02	2.0	2.41	3.75
G_{10}	1.0	0.98	−2.0	0.00	0.39
G_{11}	1.0	1.04	4.0	0.00	−0.37
G_{12}	1.0	1.01	1.0	2.73	1.32
G_{13}	1.0	0.99	−1.0	0.00	−0.26
G_{14}	1.0	0.98	−2.0	0.00	−0.50
G_{15}	1.0	1.02	2.0	0.00	−0.05
G_{16}	1.0	0.96	−4.0	−3.36	−2.67
G_{17}	1.0	1.02	2.0	0.00	−0.61
G_{18}	1.0	0.50	−50.0[a]	−50.09	−47.33
G_{19}	1.0	0.98	−2.0	−1.41	−3.81
G_{20}	1.0	0.96	−4.0	−4.40	−4.72

[a] Faults.

the elements. The penalty multipliers β_t in (26) are set to 1000, sufficiently large to ensure that the nonlinear constraints (circuit equations) are satisfied.

We tested this example for four other different starting points. The Huber approach correctly located the faults in all the cases. The ℓ_1 method was successful in three of the cases, but failed in one of the cases (trapped in a different local minimum).

VII. ONE-SIDED HUBER OPTIMIZATION FOR CIRCUIT DESIGN

In a large-scale design problem, we often wish to optimize a small number of dominant variables in order to obtain a good starting point for the following full-scale optimization.

We consider a five-channel 12 GHz multiplexer with a total of 75 optimizable variables including waveguide manifold spacings, channel filter coefficients, and input/output couplings [18]. We know that the multiplexer responses are highly sensitive to the spacing lengths, which are initially set to half the wavelength corresponding to the channel center frequencies. The common port return loss and individual channel insertion loss responses at the starting point are shown in Fig. 10.

We first try to optimize a small number of dominant variables. We select the spacings and the channel input transformer ratios (10 variables) and consider a lower specification of 20 dB on the common port return loss. The minimax solution with these variables is shown in Fig. 11 and the one-sided Huber solution is shown in Fig. 12. The worst-case errors in these two figures are similar. Since the worst-case errors cannot be further reduced by changing the selected variables, the minimax optimizer gains nothing from directing effort elsewhere. Using one-sided Huber optimization, on the other hand, we were able

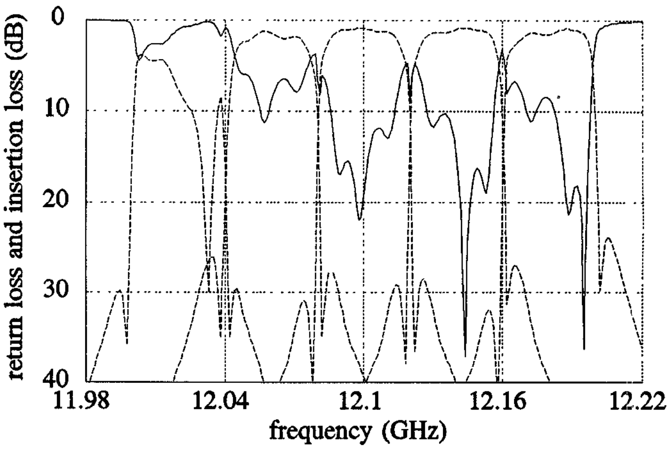


Fig. 10. Multiplexer responses at the starting point, showing the common port return loss (solid line) and the individual channel insertion losses (dashed line).

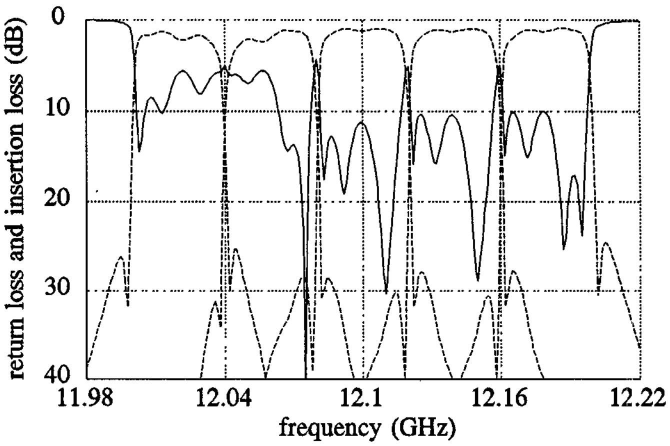


Fig. 11. Multiplexer responses after the minimax optimization with 10 variables: spacings and channel input transformer ratios; the common port return loss (solid line) and the individual channel insertion losses (dashed line). This result hardly improved upon the starting point shown in Fig. 10.

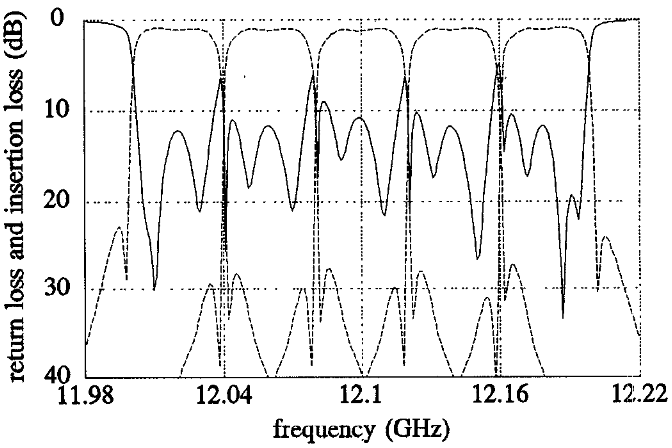


Fig. 12. Multiplexer responses after the one-sided Huber optimization with 10 variables: spacings and channel input transformer ratios; the common port return loss (solid line) and the individual channel insertion losses (dashed line). This result is significantly better than the minimax solution of Fig. 11.

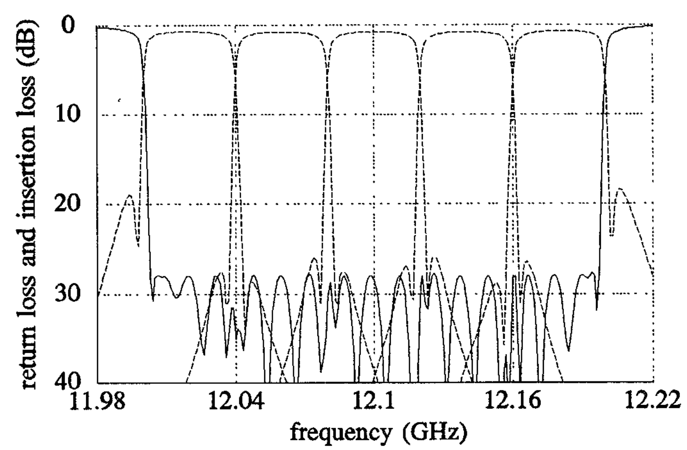


Fig. 13. Multiplexer responses after the minimax optimization with the full set of 75 variables, showing the common port return loss (solid line) and the individual channel insertion losses (dashed line).

to obtain a good starting point for subsequent optimization. The one-sided Huber optimization took 28 min on a Sun SPARCstation 1+.

From the solution shown in Fig. 12, we increase the number of variables from 10 to 45, include an upper specification of 2 dB on the channel insertion loss, and restart the one-sided Huber optimization. Then a minimax optimization with the full set of 75 variables is performed, resulting in the multiplexer responses shown in Fig. 13.

VIII. Comparison of Dedicated and Generic Algorithms

Since the Huber objective function is continuous and has a continuous gradient, it may be tempting to conclude that it is a straightforward matter to formulate the objective function and then minimize it by a generic algorithm, such as a quasi-Newton or direct search method.

We conducted a comparison between our dedicated algorithm (Section III) and three generic algorithms available in the OSA90/hope$^{\text{TM}}$ system: quasi-Newton, conjugate gradient, and simplex search.

The first test case is to estimate the mean value of the FET parameter τ as described in Section V. Only one variable is involved in this case, and all the algorithms under test converged to the correct solution. Table IV lists the number of function evaluations required by each algorithm from four different starting points. It shows that our dedicated Huber algorithm is more efficient than the generic ones.

We also attempted to apply the generic algorithms to the data fitting problem of Section IV, which involves four variables. None of them is able to find the correct solution unless starting very close to the solution. It attests to the need for the dedicated algorithm for solving multidimensional problems.

As derived in Section II, the Hessian of the Huber objective function is discontinuous wherever one of the error functions (f_j) crosses the threshold value. This may pose a serious problem for generic algorithms that explicitly rely on the continuity of the Hessian matrix.

TABLE IV
Number of Function Evaluations Required by Different Algorithms[a]

Algorithm	Starting Point			
	1.5	2	2.25	3
Dedicated Huber	4	4	4	4
Quasi-Newton	8	5	5	7
Conjugate-gradient	13	13	11	14
Simplex	26	16	16	24

[a] The optimization problem is to estimate the mean of FET parameter τ using the Huber objective function.

IX. Conclusions

We have introduced the unique Huber concept and presented novel results for analog circuit CAD. We have demonstrated that the Huber concept is consistent with practical engineering intuition. It should have a profound impact on modeling, design, fault diagnosis, and statistical processing of circuits and devices. We have exploited the robustness of Huber optimization, supported by strong numerical evidence. The similarities and differences between the Huber and ℓ_1, ℓ_2 and minimax objective functions have been discussed in a practical context. We have created the one-sided Huber function as an extension to accommodate upper and lower specifications in circuit optimization. A dedicated algorithm for Huber optimization has been presented. It has been shown by comparison to be more effective and efficient than generic minimization algorithms.

References

[1] J. W. Bandler, W. Kellermann, and K. Madsen, "A nonlinear ℓ_1 optimization algorithm for design, modeling and diagnosis of networks," *IEEE Trans. Circuits Syst.*, vol. CAS-34, pp. 174–181, 1987.

[2] J. W. Bandler, S. H. Chen, and S. Daijavad, "Microwave device modeling using efficient ℓ_1 optimization: A novel approach," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-34, 1986, pp. 1282–1293, 1986.

[3] P. Huber, *Robust Statistics.* New York: Wiley, 1981

[4] G. Li and K. Madsen, "Robust nonlinear data fitting," in *Numerical Analysis 1987*, D. F. Griffiths and G. A. Watson, Eds., Pitman Research Notes in Mathematics Series 170. Harlow, Essex, UK: Longman, 1988, pp. 176–191.

[5] H. Ekblom and K. Madsen, "Algorithms for nonlinear Huber estimation," *BIT*, vol. 29, pp. 60–76, 1989.

[6] J. W. Bandler and Q. J. Zhang, "An automatic decomposition approach to optimization of large microwave systems," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-35, pp. 1231–1239, 1987.

[7] A. Sangiovanni-Vincentelli, L. K. Chen, and L. O. Chua, "An efficient heuristic cluster algorithm for tearing large-scale networks," *IEEE Trans. Circuits Syst.*, vol. CAS-24, pp. 709–717, 1977.

[8] *OSA90/hope*$^{\text{TM}}$, Optimization Systems Associates Inc., P.O. Box 8083, Dundas, Ont., Canada L9H 5E7, 1993.

[9] K. Madsen and H. B. Nielsen, "Finite algorithms for robust linear regression," *BIT*, vol. 30, pp. 682–699, 1990.

[10] J. J. Moré, "Recent developments in algorithms and software for trust region methods," in *Mathematical Programming, the State of the Art.* Bonn, West Germany: Springer-Verlag, 1982, pp. 258–287.

[11] J. W. Bandler, R. M. Biernacki, S. H. Chen, J. Song, S. Ye, and Q. J. Zhang, "Statistical modeling of GaAs MESFETs," in *IEEE MTT-S Int. Microwave Symp. Dig.*, Boston, MA, 1991, pp. 87–90.

[12] J. W. Bandler, R. M. Biernacki, Q. Cai, S. H. Chen, S. Ye, and Q. J. Zhang, "Integrated physics-oriented statistical modeling, simula-

tion and optimization," *IEEE Trans. Microwave Theory Tech.*, vol. 40, pp. 1374–1400, 1992.

[13] *HarPE*™, Optimization Systems Associates Inc., P.O. Box 8083, Dundas, Ont., Canada L9H 5E7, 1993.

[14] Measurement data provided by Plessey Research Caswell Ltd., Caswell, Towcester, Northamptonshire, England NN12 8EQ, 1990.

[15] A. Materka and T. Kacprzak, "Computer calculation of large-signal GaAs FET amplifier characteristics," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-33, pp. 129–135, 1985.

[16] J. W. Bandler, R. M. Biernacki, A. E. Salama, and J. A. Starzyk, "Fault isolation in linear analog circuits using the ℓ_1 norm," in *Proc. IEEE Int. Symp. Circuits Syst.*, Rome, Italy, 1982, pp. 1140–1143.

[17] J. W. Bandler and A. E. Salama, "Fault diagnosis of analog circuits," *Proc. IEEE*, vol. 73, pp. 1279–1325, 1985.

[18] R. M. Biernacki, J. W. Bandler, J. Song, and Q. J. Zhang, "Efficient quadratic approximation for statistical design," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 1449–1454, 1989.

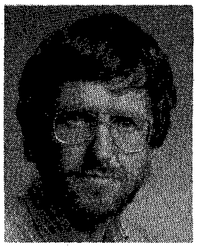
John W. Bandler (S'66–M'66–SM'74–F'78), for a photograph and biography, see this issue, p. 2278.

Shao Hua Chen (S'84–M'88), for a photograph and biography, see this issue, p. 2278.

Radoslaw M. Biernacki (M'85–SM'86), for a photograph and biography, see this issue, p. 2278.

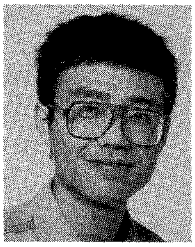
Li Gao was born in China in 1955. She received the B.S. degree in mathematics from Shandong University, China, in 1982 and the Ph.D. degree from the Institute for Numerical Analysis, Technical University of Denmark, Lyngby, Denmark, in 1988.

From 1988 to 1989, she carried out postdoctoral work in the Department of Mathematics, Peking University, China. Since 1989, she has been working as an Associate Professor at Peking University. Her research interests include optimization methods, robust parameter estimation and interval analysis.



Kaj Madsen was born in Denmark in 1943. He received the Cand.Sci. degree in mathematics from the University of Aarhus in 1968 and the Dr.Techn. degree from the Technical University of Denmark in 1986.

From 1968 to 1988 he was a Lecturer in Numerical Analysis, apart from the academic year 1973 to 1974, when he was with AERE Harwell, Didcot, England. Much of his career has been spent with the Institute for Numerical Analaysis, Technical University of Denmark, but during the years 1981–1983 he was with the Computer Science Department, Copenhagen University. During the summer of 1978, he visited McMaster University, Hamilton, Ont., Canada. He has been a frequent visitor since that time. In 1988 he became Professor with the Institute for Numerical Analysis, Technical University of Denmark. His fields of interest in teaching and research are programming languages, optimization, and interval analysis.



Huanyu Yu was born in Beijing, China, on May 27, 1966. He received the B.Eng. degree in electronic engineering from Tsinghua University, Beijing, China, in 1990, and completed the M.Eng. degree program in electrical and computer engineering from McMaster University, Hamilton, Ont., Canada in 1993.

From July 1990 to May 1991, he was a research associate with the Information System and Computer Application Group, Department of Electronic Engineering, Tsinghua University. He joined the Simulation Optimization Systems Research Laboratory and the Department of Electrical and Computer Engineering, McMaster University, Hamilton, Ont., Canada, in September 1991, where he is currently a teaching assistant and graduate student working toward the Ph.D. degree. He has been awarded an Ontario Graduate Scholarship for the academic year 1993–1994. His research interests include optimization theory and its circuit design application, computer graphics and user interfaces for CAD software.